Learning to Un-Rank: Quantifying Search Exposure for Users in Online Communities

Anonymous Authors
Anonymous Affiliation
Affiliation Address
anonymous@email.com

ABSTRACT

Search engines in online communities such as Twitter or Facebook not only return matching posts, but also provide links to the profiles of the authors. Thus, when a user appears in the top-*k* results for a sensitive keyword query, she becomes widely exposed in a sensitive context. The effects of such exposure can result in a serious privacy violation, ranging from embarrassment all the way to becoming a victim of organizational discrimination.

In this paper, we propose the first model for quantifying search exposure on the service provider side, casting it into a reverse k-nearest-neighbor problem. Moreover, since a single user can be exposed by a large number of queries, we also devise a learning-to-rank method for identifying the most critical queries and thus making the warnings user-friendly. We develop efficient algorithms, and present experiments with a large number of user profiles from Twitter that demonstrate the practical viability and effectiveness of our framework.

KEYWORDS

Search Exposure, Ranking Exposure, Privacy, Information Retrieval

ACM Reference format:
Anonymous Authors. 2016. Learning to Un-Rank:
Quantifying Search Exposure for Users in Online Communities. In *Proceedings of , , (CIKM 2017),* 10 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Motivation: When using a search engine to query social media, such as Twitter or Facebook, the system not only returns matching posts, but also identifies the users who have written these posts. This *search exposure risk* is particularly pronounced when a user's post appears in the top-k results for a sensitive keyword query.

Note that exposure is different from just having contents visible within social media. When Facebook introduced the *News Feed* feature, a lot of users responded with outrage. They felt their privacy was being violated, even though the new feature only meant that newly generated content would be broadcasted upon log-in to whomever would have access to that content anyway [?]. Analogously, in the context of search systems, while a user may be fine with posting about controversial political issues or using swearwords, she may feel very uncomfortable with these posts being returned as top-ranked results. Content found this way could be used, for example, in stories written by journalists or bloggers, and attract uninvited attention to the user's account. There are also risks regarding search exposure by unique strings. An adversary with a list of generated credit card numbers could issue these as queries to find other personal information necessary for credit card transactions. An adversary with a list of e-mails could issue these to find answers to security questions necessary to reset passwords.

State of the Art and Limitations: Despite the existence of such threats, to the best of our knowledge, there is no support for users to find out about their search exposure risks. The only way would be to try out all possible queries and inspect their top-k results, yet this is all but practical. The service providers – search engines or social network platforms – do not provide such support at all.

Work in the broad area of exposure has been tangibly motivated by a study showing the big discrepancy between the expected and actual audience of user-generated content [?]. Exposure has so far been addressed in other contexts, such as information exposure among friends in social networks [10], location exposure [16], longitudinal information exposure [11], controlling information sharing [15], or exposure with respect to sensitive topics [3]. The importance of exposure control has led service providers to introduce features such as Facebook's *View As*, which informs a user how her profile appears to other people. However, this does not quantify the exposure, and the problem of search exposure in particular has been disregarded completely.

Problem and Challenges: To the best of our knowledge, this paper is the first to address the problem of modeling, analyzing and quantifying search exposure risks. As the risk is most significant when a user is spotted in the top-k results of a keyword query, with k typically being 10, our goal is to identify, for each user, those queries. This information can then be used to guide the user, for example, in deleting posts or restricting their visibility. In an online setting, a tool based on our model could even alert the user about the exposure before submitting a post.

This problem poses a number of challenges:

- *Efficiency:* A user could possibly be found by millions (if not billions) of distinct queries. An algorithm to identify the critical queries thus faces a huge scalability and efficiency problem.
- *Dynamics:* With the high rate of new online contents, the critical queries cannot simply be computed offline from a query log. The exposure of a user keeps continuously shifting.

CIKM 2017,
© 2016 ACM. 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

- *Usability:* Showing all queries for which a user appears in the top-k results would, in many cases, drown the user with millions of irrelevant or merely ”small risk” queries and miss the point of guiding the user. Thus, it is crucial that the queries are ranked by an informative measure of, possibly user-specific, sensitivity.

From a user perspective, reducing search exposure can be seen as a problem of *“inverse search engine optimization”*, inverse SEO for short. SEO aims to push a user to the top-ranked results for certain queries. Here, the goal is the opposite: users want to be moved to the low-ranked tail of answers, or even be removed from the search results of particularly sensitive queries.

Approach and Contributions: Keyword search with answer ranking can be seen as the problem of finding the top-k nearest-neighbor posts according to a given similarity function. Thus, we can model search exposure as a *reverse k-nearest-neighbor* problem (RkNN): for each post, we want to find the queries for which the post is among the top-k results. To identify these queries, we develop an efficient algorithm that builds on threshold-based pruning [**?**].

To assist a user in understanding her search exposure risks, we devise measure for ranking the queries in the user’s RkNN set, which is potentially a huge set with millions of queries. To this end, we devise informative measures that combine topical sensitivity (e.g., usually higher for health topics that for posts about movies), query selectivity and entropy (e.g., higher for queries about birth dates, social security numbers, etc.), and user surprisal (e.g., high for posts about a user’s children as opposed to her professional background).

The salient contributions of this paper are:

- modelling the search exposure problem;
- an efficient algorithm for computing the RkNN set of queries for which a user appears in the top-k results;
- informative measures for ranking queries towards identifying the truly sensitive ones;
- an experimental study with a large set of Twitter profiles, providing insights on the RkNN sets and the effectiveness of our query ranking methods.

2 PROBLEM STATEMENT

Preliminaries. Assume we have a set of users U and a set of documents D posted by the users. We denote the fact that a post d is written by the user u by $d \in u$. The profile of each user is defined as the set of all documents she has posted in a given community.

Search exposure. The problem of search exposure of user u can be then rephrased as finding all the reverse k-nearest neighbors of u, i.e, the set of all the queries for which any of the posts of u comes among the top k results. We call the sets of such queries as *exposure sets.*

Generation of exposure sets. Before computing the exposure sets for each user, we first compute these for each document d as follows:

$$RkNNs(d) = \{(q, r) | q \in Q \land d \text{ is the } r\text{th NN of q}\} \tag{1}$$

where Q is the set of all queries and r is the rank of d for the query q ($r = rank(q, d)$). According to Emrich et al., the above equation is equivalent to the definition of a bichromatic RkNN [8].

Accordingly, we define the exposure set of each user as the union of the exposure sets of all the documents in her profile. We denote the exposure set of the user u by $RkNNs(u)$ which is defined as follows:

$$RkNNs(u) = \{(d, q, r) | (q, r) \in RkNNs(d) \land d \in u\} \tag{2}$$

We discuss efficient generation of exposure sets in Sec. 3.

Ranking of queries in exposure sets. Exposure sets of prolific users might be big and dominated by rare, non-informative, or non-critical queries. On the other hand, exposure by certain sensitive queries might leave the user uncomfortable. Therefore, to make the exposure sets user-friendly, we want to rank the triples in $RkNNs(u)$ such that *the queries the users would not want to be searched by appear at the top.* This defines the notion of relevance in our ranking problem, termed *search exposure relevance.* We discuss the exposure set ranking methods and our notion of relevance in Sec. 4.

3 GENERATING EXPOSURE SETS

Ranking model: Generating the exposure set (set of RkNNs) requires the knowledge of the ranking model. In this work, we assume the documents are ranked using the standard technique of language models. More specifically, given a query q the relevance of the document d is measured by the likelihood of model d generating q. Assuming no dependency between terms, the logarithmic query likelihood is computed as follows [**?**]:

$$\rho(q = t_1 t_2 ... t_n, d) = \sum_{i=1}^{n} \log \frac{f_{t_i,d} + \mu \frac{c_{t_i}}{|C|}}{|d| + \mu}, \tag{3}$$

where $f_{t_i,d}$ is the frequency of the ith term of q in d, c_{t_i} is the frequency of the term t_i in the whole corpus, $|C|$ is the total number of terms in the corpus and μ is the Dirichlet smoothing coefficient.
Queries: Any combination of words is a valid query. However, considering all the possible queries makes our problem intractable. Therefore, we limit the set of queries to those that have at least one exact match in our data set. In other words, for each query $q = t_i, t_{i+1}, ...t_{i+l-1}$ there should be at least one document in the dataset that contains all the terms t_1, t_2, ... t_l . The resulting set of queries, however, is still very large as it contains all the possible combinations of terms in every document. According to [**?**], the average length of queries issued on twitter is 1.64 words. Therefore, we add the second constraint on the length of the queries to be at most two.
Approach: Informing the users about the exposing queries can be either retroactive or preventive. In the retroactive approach, users are notified about their exposed documents posted so far. Preventive warnings, however, make users aware of exposure risk at the time of uploading a document. Depending on the approach chosen, a service provider may opt for static or dynamic generation of RkNNs. In the former case, the exposure set of the users can be computed statically and fetched upon user’s request. However, as the insertion of the new documents or the deletion or the update of

already existing ones may affect the exposure set of the users, the recomputation of the RkNNs would be necessary every so often. On the other hand, preventive warnings require dynamic generation of RkNNs for a document in an efficient way.

The problem of **B**ichromatic **R**everse k **N**earest **N**eighbors (BRkNNs) has been extensively studied so far. However, there is no generic efficient solution for this problem as the efficiency is achieved through applying various filtering techniques that are mainly dependent on the distance (or similarity) function. For example, for the models with the euclidean distance (or the cosine similarity) as the distance (or similarity) measure, many filtering methods have been suggested based on the geometric properties [14**? ?**]. However, for the models with textual queries only a few works exist that cannot be fully applied to our problem as either the setting is monochromatic [**?**], the indexing tree of the queries should be built [**?**] which is not scalable for very high demensions, the storage of k nearest neighbors for all the queries are required [7] where the approach is not flexible with different values of k or only the conjunctive queries are considered [1]. In the rest of this section, we present our suggested algorithm for the dynamic generation of the RkNNs for the textual data.

In our model, it can be shown that finding an exposing query of length l ($q = t_i t_{i+1} ... t_{i+l-1}$) for a particular document d where d is the nearest neighbor of q is NP-hard (see the Appendix. A).Although limiting the set of queries to bigrams turns our problem to a polynomial one, the query search space is still large. Therefore, we introduce the "RkNN Growth" algorithm for more precise pruning of the search space.

RkNN Growth Algorithm (RGA): The underlying idea of this algorithm is to use the information on the one-word RkNNs to find the two-word RkNNs. In this algorithm, we first start with finding the unigram RkNNs for the given document d. For this, we build the inverted index and sort the posting list L_{t_i} of each term t_i based on the relevance of the documents (posts) to t_i. Now, we can compare the score of the kth document in L_{t_i} with $\rho(q = t_i, d)$ in constant time to decide whether the term t_i is among the RkNNs of the document d or not.

In order to find the bigram RkNNs, we label the terms appearing in the document d as follows:

- "T" which stands for Top, if the term appears among the RkNNs of document d.
- "NT" which stands for Not Top, if the term is not an RkNN of the document d

We refer to the other terms as "NE" (Not Existing) as they do not appear in the document d. According to this labeling, there are five possible labeling for the two-word queries (T-T, T-NT, T-NE, NT-NT, NT-NE). Note that due to the disjunctive nature of the queries, labels T-NE and NT-NE are also allowed. It is straightforward to show that the two following propositions hold in our model:

Proposition 1. *If $rank(t_i, d) = k'$ and $rank(t_j, d) = k''$ and $k' + k'' < k$, then the query $q = t_i t_j$ is an RkNN of the document d.*

Proof. According to the monotonicity of our ranking model, for a document d' to be more relevant to the query $q = t_i t_j$ than d, either $\rho(t_i, d') > \rho(t_i, d)$ or $\rho(t_j, d') > \rho(t_j, d)$ should hold. However, these conditions hold for less than k documents in total as $k' + k'' < k$. □

Proposition 2. *If the term t_i in the document d has the label "NT" and the term t_j has label "NE", then the query $q = t_i t_j$ cannot be an RkNN of the document d if there are at least k documents that are both shorter and more relevant to the term t_i than the document d.*

Proof. As the relevance score is negatively proportional to the length of the documents, the above statement can be interpreted as the document d being dominated by more than k other documents in terms of relevance to both the terms t_i and t_j. Therefore, d cannot have $q = t_i t_j$ as an RkNN. □

We use the above statements to prune the query search space. For the remaining queries, we run the Reverse Threshold Algorithm (RTA) introduced by Vlachou et al. [17].

Algorithm 1 illustrates the steps of RGA. For each token t_i in the input document d, first the set of all candidate queries (C) containing t_i is extracted (Line 2). As all of the candidate queries contain the term t_i, we initiate the buffer for the reverse threshold algorithm (RTA) with the k nearest neighbors of t_i (Line 3). We perform the first step of pruning (if possible) in Line 5 by removing all the possible extensions of type "NT-NE". We show in Sec. 5 that this step prunes a huge number of candidate queries. Next, we iterate over the set C and based on the label of t_i we test the applicability of Proposition 1 or Proposition 2 for query pruning (Line 6, 9). The remaining queries are passed to the function RTA where the queries are either filtered or added to the exposure set of d (Line 12). Finally, the exposure set of the document d is reported.

Algorithm 1: RkNN Growth Algorithm

```
Input  :Document $d$, the value of $k$, the set of queries $Q$
Output :$RkNNs(d)$
1  foreach token $t_i \in d$ do
2      Compute the candidate set of queries $C$
3      Place the $k$ nearest neighbors of $q = t_i$ in $buffer$
4      if $t_i$ has label "NT" and the Proposition 2 holds then
5          Remove all the queries $q = t_i t_j$ of type "NT-NE" from
           $C$
6      end
7      foreach $q \in C$ do
8          if $t_i$ has label "T" then
9              Check whether $q$ can be pruned by Proposition 1
10         end
11     end
12     Run RTA for the remaining queries
13 end
14 Report R$k$NNs of $d$
```

4 RANKING OF QUERIES IN EXPOSURE SETS

Generating the exposure sets is not enough for the results to be presentable to ends users for two reasons. First of all, for many users the size of their *RkNN* set is simply be too big for easy consumption.

Our experiments on a sample of $50K$ user profiles from Twitter later confirms this – more than $35K$ users are exposed by more than 100 queries, even when only unigram and bigram queries are considered. Figure 1 shows the distribution of exposure set size for users from the sample.

Moreover, since we don't a priori exclude queries such as infrequent or numerical tokens most RkNN sets will end up dominated by garbage queries. Leaving such queries in during the generation phase is a design choice motivated by the 'worst case scenario' principle than often guides privacy and security research. While most users will find these queries uninformative, for some people it might be important to know they are searchable by certain URLs (e.g., when the domain is known to contain sensitive content) or numbers (e.g., their year of birth or the prices od products they buy). Table 1 shows examples of raw exposure sets where queries are ordered by the rank position of the corresponding user post. These examples illustrate the need for ranking the queries before presentation to end users.

4.1 Learning to rank the exposing queries

Recall the search exposure sets defined by Eq. 2. We want to rank the triples within these sets according to search exposure relevance, i.e., such that the queries the users do not want to be searched by appear at the top. The traditional IR learning to rank setup, in which the learned function orders the documents by relevance to queries, is replaced by one where we rank the queries according to relevance to users.

Each user-document-query triple can be represented as a feature vector $\Phi(u, d, q)$. For each user, together with the relevance score annotation, these form partial rankings r_u determining pairwise relevance constraints between the tuples in the form:

$$(d_u^{i_1}, q^{j_1}) >_{r_u} (d_u^{i_2}, q^{j_2}).$$

We want to learn a ranking function $r^*\Phi(u, d, q)$ that minimizes a loss measure over the partial training rankings. For example, when learning to rank using SVM^{rank} [?], it is the number of violated pairwise constraints that is minimized, which implicitly leads to maximization of Kendall's τ between the training and learned rankings.

We describe the features and relevance scores we used to learn the ranking function in the following two sections.

4.2 Features

4.2.1 *Semantic features.*

The meaning of words certainly plays an important role in determining criticality in the context of search exposure. Topical sensitivity has also been proven useful in the context of privacy risk quantification from text in user studies [3]. To capture the coarse-grained semantics of the queries, we annotate them with categories from the LIWC dictionaries [?]. LIWC categorizes words into 80 linguistically and psychologically meaningful categories such as *positive emotion* (love, nice, sweet), *affective processes* (happy, cry, abandon), *swear words* (damn, piss, fuck), *anxiety* (worried, fearful, nervous), or *sexual* (horny, love, incest). We create one binary feature based on each category, with a value of 1 if any of the query words matches any of the words from the category.

4.2.2 *Uniqueness of queries.*

While any query generated from a community's text contents search-exposes some of its members, from the perspective of a single user, these are the rare tokens that are more likely to lead to exposure. While a considerable portion of rare queries are simply meaningless noise, it is possible that there are meaningful infrequent tokens with the potential of privacy violation. Recall some of our motivating examples where an adversary searches for information associated with a given sensitive domain, or an e-mail address.

We propose two features that capture how rare a query is: query selectivity and query entropy. We define the query selectivity as the number of documents matching the query exactly:

$$selectivity(q) = |\{d : q \in d\}| \tag{4}$$

This measure will be low for queries which appear infrequently.

Another aspect of a query being unique for a user is how skewed the distribution of the relevance scores is. We capture this by measuring the entropy over the distribution of ranking scores of the top-10 returned results. Let R be the distribution of the relevance scores of the top-k results. We measure the entropy of the query as:

$$entropy(q) = H(R), \text{ where } R(i) = \frac{score(q, d_i)}{\sum_j^k score(q, d_j)} \tag{5}$$

Note that these measures are not dependent on a given user, but are dependent on the community as a whole, i.e., the relative rankings of queries in different communities might differ. For instance, while the query *Lyme borreliosis* might be an infrequent query on Twitter, it could me more popular in a medical Q&A forum.

4.2.3 *User surprisal.*

The lexical context of a user might also matter when determining the criticality of a query. Imagine a user with a Twitter profile where she posts mostly professional content. It would not be surprising, and perhaps even desirable, that the userfis posts are returned as top results to the queries from that professional domain. However, if it turns out that the user profile comes up at the top only to the query *funny cats* that matches that single post the user has ever made outside of the professional domain, this might be both unexpected and undesirable.

We propose to capture this intuition using surprisal, which is measured by reversing the probability of the query being generated from a user's vocabulary distribution estimated from the posts:

$$surprisal(q, u) = log\left(\frac{1}{P(q|u)}\right) = log\left(\frac{1}{\prod_{w \in q} P(w|u)}\right) \tag{6}$$

To account for the sparsity of user profiles, we compute these probabilities using Dirichlet smoothing. This feature could help rank higher a query *funny cats* that matches a document in an otherwise professional Twitter profile.

4.2.4 *Post surprisal.*

Even though these are the queries that are ranked, the users might not want to be matched to a non-critical query when it exposes a critical post. Similarly to surprisal of queries, we define the surprisal of posts that are matched by the exposing queries by replacing q by d in Eq. 6.

4.2.5 *RkNN features.*

Two traditionally considered ways for ranking the reverse nearest neighbors by relevance to the user are the similarity of the reverse neighbor to the user and the rank of the

Table 1: Examples of raw (unranked) exposure sets.

aim oshtitsbaj,	asleep oshtitsbaj,	http://ra*.com/teh_ba splash,	mac vanilla,	suck wake,	mood sick	
emma sun,	watch xxxxxx,	forget toast,	@jeff* fall,	heavyweight ladder,	omg tan,	alcohol nice
blown death,	comin lake,	parilla wait,	bathroom wanna,	crush hannah,	friend lord,	record woman

reverse neighbor. While not likely to be useful when the relevance is defined as criticality, we include these features for comparison. We measure similarity using the probability of generating the query from the posting history of u:

$$similarity(q, u) = log\,(P(q|u)) \tag{7}$$

Let d_u be the post of a user u that is returned as an answer to query q at position $rank(q, d_u)$:

$$rankposition(q, u) = rank(q, d_u) \tag{8}$$

4.2.6 Syntactic features. We also introduce a number of binary post-dependent features that characterize emotional display or content the users might not want exposed through search. These include:

- *has_url* (set to 1 if the post contains a URL),
- *has_at_mention* (set to 1 if the tweet mentions another user),
- *has_hashtag* (set to 1 of the query contains a hashtag).
- *has_emoticon* (set to 1 if the post contains an emoticon),
- *has_repeated_punctuation* (set to 1 is any token in the post ends with a double exclamation mark, double question mark, or an ellipsis),
- *has_repeated_vowels* (set to 1 if any token in a post contains a vowel repeated at least three times in a row),
- *has_laughter* (set to 1 if any token contains a substring like *haha*, with different vowels).

4.3 Relevance

Search exposure relevance differs in many ways from the topical relevance of traditional IR tasks. A query might be relevant not only because it's topically sensitive, but also because it could embarrass, offend, or otherwise violate the privacy of the exposed person. The subjective nature of such judgments makes the manual collection of relevance at scale an extremely time-consuming task, especially if done by external evaluators. To decide which queries would be relevant, a judge would have to put themselves in the shoes of the evaluated user, imagine who that person is based on the contents of the profile, and decide which queries would concern her. Moreover, a judge would have to come up with likely threat scenarios. It is a non-trivial task to prime the judges regarding these issues without biasing them. With all these considerations, we derive implicit relevance scores from other user-generated signals that indicate reluctance to be associated with a given content. Implicit relevance signals, especially in the form of clickthrough patterns, are commonly used in traditional retrieval tasks [**?**]. The remainder of this section presents our method for synthesizing the search exposure relevance scores.

User score. If a user deletes a post, it is the most direct signal she does not want to be associated with its content. Thus, a query matching a post that got deleted after publication receives a user score of 1, whereas queries matching non-deleted posts receive a user score of 0. While a service provider quantifying exposure would have a direct access to this information, there are also ways for collecting it outside of the system [11]. We describe our collection method in more detail in the experimental section.

Community score. The deletion information is a noisy signal, however, as users delete posts for a variety of reasons, including language or double posting errors. We want to sanitize these scores using stronger, community-wide signals that encode the differences in language distributions in anonymous and non-anonymous communities. These linguistic differences have been observed for instance when comparing posts from Twitter and Whisper (an anonymous microposting platform) [**?**]. Having estimated the vocabulary distributions in an anonymous (P_{anon}) and a non-anonymous ($P_{non-anon}$) community, we treat the relative probability of a query being generated from these distributions as a community-wide signal that users do not want to be associated with the keywords: More precisely, we set the community score of a query to:

$$community\_score(q) = \frac{P_{anon}(q)}{P_{non-anon}(q)} = \prod_{w \in q} \frac{P_{anon}(w)}{P_{non-anon}(w)}$$

Golden ranking. Finally, we derive the relevance as a linear combination of both scores:

$$score(q) = \alpha \cdot user\_score(q) + (1 - \alpha) \cdot community\_score(q)$$

Combining both scores allows us to discount the relevance of noisy queries that match deleted posts, as well as add relevance to sensitive queries matching posts that did not get deleted, as the user perhaps did not have any privacy concerns in mind.

5 EXPERIMENTS

5.1 Dataset

For our experiments we used a set of Twitter profiles from the longitudinal exposure study by Mondal et al. [11]. It consists of 51,550 users with a total of about 5.5 million tweets, collected over the year 2009.

5.2 R*k*NN Generation

5.2.1 Data Preparation. : Before generating the RkNNs, we performed data cleaning, including stop word removal, lemmatization and stemming. As a result, we extracted around 2 million unique tokens. For the query generation, we considered all unigrams and bigrams appearing in at least one tweet which counted around 45 million in total.

5.2.2 Static generation. In order to report the exposure set of all the users, we found the k nearest neighbors of each query using

Query Type	Average number of candidate queries	Queries present in the exposure set (%)
T-T	0.5	100
T-NT	4.6	99
NT-NT	15.5	73
T-NE	348.6	90
NT-NE	125112.1	0.4

Table 2: Statistics on query types.

the threshold algorithm with 16 parallel processes and added the query to the exposure set of the resulting tweets' authors. Using a cluster node with 1.5 TB memory and 48 CPU cores it took about 10 hours in total to generate the exposure set of all the users.

5.2.3 Dynamic generation. : In Section 3, we introduced four types of queries. Table 2 shows the average number of candidate queries of each type per tweet and also the average percentage of each type appearing in the exposure set of a tweet. The values in the last row confirms the importance of pruning based on 2 in RkNN Growth Algorithm (RGA) as only 0.4 percent of the candidate queries of type "NT-NE" finally end up in the exposure set.

We experimented RGA on a random sample of 1000 tweets. Our experiments showed that using Proposition 2 we could prune all the possible extensions of about 87% of one-word queries of type NT to queries of type $NT - NE$ for each document d. More precisely, if the document d is not among the top k results of the query $q = t_i$, then with probability of at least 0.87 the document d would not be among the top k results of any query $q = t_i t_j$ with type NT-NE". As a result, RGA pruned 99% (96% through Proposition 2 and 3% through RTA) of all the candidate queries of type NT-NE" for a document on average.

5.3 User RkNNs Analysis

Figure 1 shows the distribution of the size of the exposure set of the users in the dataset for different values of k. As can be seen in Figure 1, there are a few users whose exposure set is huge (in order of millions). These users are called hubs. In other words, the profile of these users appear among the top results of so many queries. On the other hand, there are 3196 users (6% of all) with empty exposure list. We call these users anti-hubs as their profile does not show up for any query.

Table 3 shows some examples of the tweets of hubs and anti-hubs in our dataset. Looking at the user with the largest exposure set, we noted that he only posted the titles of the songs broadcasting on the radio. Therefore, there is nothing personal in his profile. However, the profile of the 7th hub is mainly personal as can be seen in the second row of the table. The last row of the Table 3 shows that a tweet may contain sensitive words ('hate animals') although it does not come among the top k results of any query.

6 EXPOSURE SET RANKING EVALUATION

In this section, we discuss our insights into the search exposure problem through evaluation of the ranking methods, as well as an analysis of user perceptions regarding exposing queries collected in an experiment on Amazon Mechanical Turk.

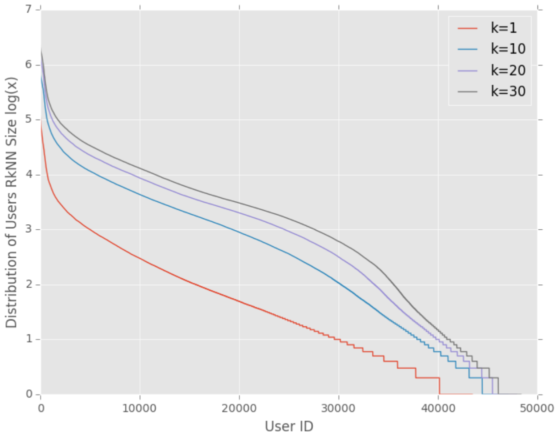

Figure 1: Distribution of the size of the exposure set for the users. The values on Y axis are in logarithmic scale with base 10.

6.1 Query ranking in exposure sets

6.1.1 Exposure sets cleaning. For the evaluation results to be meaningful, we exclude the following queries from the exposure sets: queries with tokens shorter than 3 letters, queries for which none of the tokens is an English word, queries with numerical tokens, urls, and references to other accounts. We also exclude users whose posts are primarily written in a language other than English. While all of these queries could be search exposure relevant in certain scenarios, it is unlikely that human judges who will evaluate the ranking outputs would be able to associate any meaning with them.

6.1.2 Relevance score statistics. We construct the relevance scores as described in Sec. 4.3. Community scores are derived from the Whisper dataset from collected by Correa et al.[?], and the Twitter dataset collected by Mondal et al. [11]. The Twitter dataset, moreover, comes with the information regarding tweet deletion. More precisely, by querying the Twitter API using a subset of the tweet IDs, the authors were able to determine which tweets got deleted after publication. This information was collected for 11M tweets, 400K of which turned out to have been removed. We use these signals as the user score.

Because the information about post deletion is limited, the ground truth provides us with only a partial ranking over RkNN queries. We therefore exlude the queries for which we cannot infer relevance from the evaluation in this part. These include: (i) the queries matching posts for which we don't have the deletion information, (ii) the queries with only a partial overlap with the source post (it might happen that a post is returned in the top-k results for a multi-word query even though not all query words appear in the post; for such queries we do not assume the deletion information signals not wanting to be associated with the words).

User type	User ID	Example of tweets	Example of query to find the tweet	Size of exposure set
Hub	19519947	Now on 50s on 5: In Dreams by Roy Orbison	Roy Orbison	3,993,702
Hub	20463153	I hate being claustrophobic, esp when I have to go into a tight space or in a cave	claustrophobic	2,809,290
Anti-hub	29664249	studying	No query	0
Anti-hub	2847248	i hate animals	No query	0

Table 3: Examples of the tweets of two hubs and two anti-hubs.

Table 4: Most important semantic features learned by the L2R model.

Feature	Example queries
Sexual	gross kiss, breast whitney, gay shirt
Humans	dumbest guy, girl xoxo, chess kid
Friend	pal wife, fellow fool, ummm honey
Anger	buy weapon, mad scientist, idiot vegetarian

6.1.3 Ranking algorithm. To learn the ranking function, we use SVM^{rank} [**?**] with the linear kernel. Parameter C is tuned on a random sample of 10% RkNN sets, and the rest of the data is used to evaluate the ranking method in a 10-fold cross-validation.

6.1.4 Feature analysis. The weights of the decision boundary vector learned using SVM with a linear kernel can be interpreted as feature importance weights. Table 4 lists the most important features learned by our model together with example queries exhibiting the features. The model captures well that the categories related to personal issues are the ones people feel more uncomfortable sharing. High importance of words related to sexuality stem from the bias of the Whisper data – a large majority of anonymous posts from this community regard sexuality. However, the methodology we propose is general enough to handle different types of anonymous contents. For instance, as an alternative, it would be possible to collect anonymous posts from Question & Answer communities such as Quora.

6.2 User-study evaluation

Because the relevance scores used for training the algorithm constitute noisy signals for search exposure relevance, we evaluate the reranked exposure sets in a user study. The leading question is whether users themselves would find the output useful, feeling that exposure by top-ranked keywords would make them feel uncomfortable. This section provides the details of the study.

6.2.1 Evaluation setup. The question we wish to answer in this part of the evaluation is how well the learned exposure sets rankings correlate with the exposure relevance perceptions of human judges. To this end, we need to sample a number of exposure sets and a number of queries from each.

User sampling: The first important thing to note is that not all of the exposure sets contain sensitive queries. To account for this and make sure we cover the sensitive users in the evaluation, we sample users non-uniformly in the following way. Queries within exposure sets are ordered by the predicted relevance scores. The score of the highest ranking query within a set can be thought of as an indicator of how sensitive the exposure set is overall (i.e., the lower the highest score, the less sensitive content there is overall). For evaluation, we choose the 50 most sensitive exposure sets, and 50 exposure sets sampled from the remaining tail with the probability proportional to the predicted relevance of the highest scoring query. We thus evaluate 100 exposure sets in total.

Query sampling: To construct assignments with reasonable workloads, we evaluate 50 queries from each of the sampled exposure sets. Having the queries ranked by the L2R method under evaluation, we choose 25 highest scoring queries (to see how useful the top of the ranking is), and 25 queries chosen uniformly randomly from the remaining tail (to control if if the head of the ranking does not miss critical queries).

6.2.2 AMT survey. Each set of 50 sampled queries was shown to 3 Amazon Mechanical Turk workers. The queries were displayed in a random order. We required that the workers have a master qualification (to ensure the quality of annotations) and are located in the USA (to prevent language misunderstanding). Upon explaining the basic pipeline of the Twitter search engine and priming the users about what exposure is, the survey asked the following question:

> *Would you feel concerned (uncomfortable, embarrassed, privacy-violated, or threatened) if your tweet was returned as one of the top answers to these search terms? (Yes/No)*

Having three people evaluate each query leads to a 4-graded relevance scale, based on how many people chose *Yes*.

Out of $5K$ evaluated queries, 10% had a score of 3, 12% had a score of 2, 24% had a score of 1, 54% had a score of 0. Inter-annotator agreement measured by Fleiss' κ was 0.376, which corresponds to a fair agreement.

6.2.3 Results. We report the values for NDCG@[5,10,20] and Kendall's τ. Since the collected scores offer good interpretability in terms of binary relevance, we also report Precision@[5,10,20] assuming a query is search exposure relevant if was marked by at least one judge.

Table 5 shows the results of the user-study evaluation. Note that, although the queries were sampled from the L2R-ranked exposure sets, the collected judgements also let us evaluate the other ranking heuristics. Majority of these perform significantly worse than the L2R method – differences significant by a paired t-test with $p < 0.05$ are marked with the $*$ symbol. The strongest heuristics include post surprisal and selectivity. Both of these quantities capture a different aspect of the rareness of the content, and thus shine in situations where, for instance, the judges thought that exposure by a typo might lead to embarrassment. We also observed that a large fraction of query tokens are typos that can be mapped to a sensitive

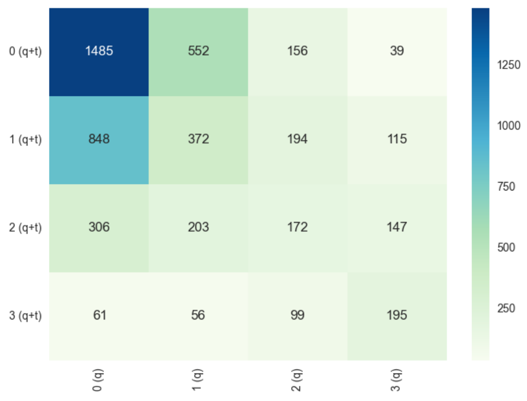

Figure 2: Influence of the tweet context on search exposure relevance (q - judgement for query only, q+t - judgement for query with tweet).

word. Such queries were often marked by the judges as relevant, and because of their rareness, heuristics such as selectivity gain in performance.

7 INSIGHTS INTO SEARCH EXPOSURE RELEVANCE

7.1 Tweet context

Another interesting question regarding search exposure relevance is whether it is influenced by the context of the returned tweet. It might happen that a query that looks sensitive is constructed from words that do not form a coherent context within a post, thus being a false alarm. On the other hand, innocent looking queries might bring out posts that do contain sensitive content otherwise.

To gain preliminary insight into this problem, we conducted a second survey on AMT, in which the workers assessed the relevance of queries knowing the tweet that is being returned as a result; the rest of the setup remained analogous. Comparison of these two surveys is summarized in Fig. 2. Existence of dark squares outside of the diagonal suggests indeed that the context might change the exposure relevance judgement. This happens both ways, suggesting that both scenarios we mentioned in the previous paragraph are plausible. We believe that investigating the factors that influence the search exposure relevance is an interesting topic for future work.

7.2 Search exposure relevance vs topical sensitivity

Topical sensitivity is a concept that has been introduced for studying privacy risks of text, in particular for quantifying R-Susceptibility (Rank-Susceptibility) in communities where user profiles consist of textual contents [3]. It measures how likely the presence of words from different topics (understood as distributions over words) leads to privacy risks, irrespective of the user or the community context. We want to understand if there is a correlation between topical sensitivity defined this way and the search exposure relevance. We thus annotate each query form our evaluation set using the crowd-sourced topical sensitivity annotations $sensitivity(t)$ collected in the R-Susceptibility paper [3]. The sensitivity of a query is defined as:

$$sensitivity(q) = \frac{1}{|q|} \sum_{w \in q} \sum_{t} sensitivity(t) \cdot P(w|t) \qquad (9)$$

where $P(w|t)$ is the probability of a word w in the topic t.

We measure the correlation between these sensitivity-annotations and the collected relevance scores using the Pearson correlation coefficient. We find a strong correlation between these scores in case of the relevance collected for queries without the tweet context (Pearson coefficient of 0.44), and a little lower correlation (Pearson coefficient of 0.32) in case of the relevance judgements for queries with the tweet context. This results reconfirms the findings from the evaluation of the L2R method – the meaning of the query is an important factor in determining search exposure relevance, and topical sensitivity is a viable alternative for implicit relevance scores.

8 RELATED WORK

Exposure. Although, to the best of our knowledge, the problem of search exposure has not been addressed in the past, there are different aspects of user and data exposure that have been studied in the prior literature. Mondal et al. proposed exposure control as an alternative solution to access control in social networks [10], and later devised solutions for longitudinal exposure control [11]. Biega et al. quantifies privacy risks for sensitive topics in rankings based on textual posts using the notion of R-Susceptibility [3]. Exposure has also been studied in the context of individual attribute leakege, such as location [16]. Another interesting problem is that of usability of exposure warnings. Example solutions include depicting the current size of content audience by the size of a displayed pair of eyes [15].

Privacy-preserving IR. Problems studied in privacy-preserving IR include sanitization of query logs prior to a release [? ?], or obfuscation of query histories through broadened or dummy queries [? ?]. A number of works also investigate the viability of personalized search under privacy constraints [? ? ? ?].

User protection and internal audits. Service providers increasingly come under close scrutiny by external organizations and observers, including journalists and researchers. This pressure encourages the SPs to perform proactive, internal audits to improve their services and infrastructure. New solutions for increased privacy are constantly introduced to mitigate the threats for users from external adversaries in services like maps [?] or e-mail []. User data itself has also been analyzed, for example, to deliver better security protections in the context of account recovery personal questions [?]. While certain ad-hoc protections are already in place also with regard to search exposure (for instance, it seems impossible to explicitly query for credit card numbers in Google, Twitter, or Facebook, since these tokens get post-processed and end up matching other numerical tokens as well), we believe there is a need for a more direct examination and protection mechanisms regarding search exposure. Beyond privacy, there are also other societal issues that press SPs to audit their services, including the

Table 5: Exposure set ranking user-study results averaged over all users. Methods marked with * perform significantly worse than L2R on a given metric (paired t-test, $p < 0.05$).

	Prec@5	Prec@10	Prec@20	NDCG@5	NDCG@10	NDCG@20	Kendall's τ
L2R	**0.636**	**0.566**	**0.509**	**0.515**	**0.496**	**0.530**	**0.107**
Surprisal	0.448*	0.449*	0.447*	0.210*	0.245*	0.316*	-0.035*
Post Surprisal	0.480*	0.471*	0.470*	0.229*	0.262*	0.350*	0.016*
Entropy	0.472*	0.489*	0.494	0.209*	0.262*	0.347*	0.026*
Selectivity	0.548*	0.508*	0.489	0.278*	0.305*	0.378*	0.037*
Rank	0.460*	0.463*	0.466*	0.204*	0.248*	0.330*	0.018*

Table 6: Top-10 critical exposing queries for example sensitive users.

blame gay, dutch gay, gay rabo, gay guy, blame dutch, suck teacher, start tht, rider tour, donald duck, attack bad
gay racist, fuckin young, deal gay, simon watchin, fuckin kinda, guy note, kind sum, live net, bcoz gay, dnt season
adopt convert, adopt religion, convert religion, convert essay, essay religion, bon convert, bon religion, bon essay, river tonight, adult love
lesbian pregnant, lesbian music, lesbian live, boat lesbian, gay norway, end lesbian, fritt gay, bell page, star trend, lesbian uuum
oooh virgin, virgin wen, video virgin, crack oooh, oooh wen, normal tom, normal smith, smith tom, swine year, outfit xoxo
gay israel, bit web, gay gunman, michael pant, hat pant, attack bit, e-mail match, israel wtf, china tale, obama recov
david queer, queer ted, asian queer, queer warhol, david folk, model race, keith york, driver rule, kind remix, jean odd
camera stick, stick tape, rep usa, china rep, stick tehran, governor tehran, governor stick, prayer tehran, prayer tehran, israel rep
background magic, lovin turn, childhood fuck, background lovin, plz stranger, confirm design, dark wild, tear tiger, creep lovin, background creep
detail obama, alex detail, alex obama, box long, bloomberg flash, dubai investor, dubai investor, june real, june real, alybi*@gmail.com investor
u.s. union, mexico union, canada union, american union, demand democrat, agenda reform, nasa obama, american borderless, nazi obama, demand overhaul

issues of fairness and bias [?], or user satisfaction of search results [?].

Search exposure can be seen as another dimension for internal audits. Along these lines, we believe more work can be done to examine which types of search queries should be blocked altogether, and which search results should be removed to protect against finding users in sensitive contexts.

(Reverse) k-nearest-neighbors problems. Finding the set of reverse k nearest neighbors (a.k.a the influence set) of a point has been studied in various contexts such as matching the user preferences to a given product [17] or the assignment of a publication to a set of existing queries [1, 7]. The setting of such problems falls into either of these categories: monochromatic or bichromatic. This classification is based on whether the points and their reverse nearest neighbors come from an identical set (monochromatic) or not (bichromatic) [8]. Our model is an instance of a bichromatic RkNN based on the fact that the set of queries and the set of the documents are not the same.

For the efficient generation of the set of RkNNs, many algorithms have been introduced which mainly differ in their approach for pruning the search space. Some of these pruning methods are the (grid-based) reverse threshold algorithm [17] and its improved version [?], branch and bound on hierarchical grid structure [18] and branch and bound on indexing trees whose nodes have either the shape of a circle or a cone [14] or a rectangle (minimum bounding rectangles) [?]. The existing algorithms, however, are not sufficiently efficient for our problem due to high dimensionality, large cardinality of the set of queries and sparsity of the query vector space.

With the increase in the size of the (R)kNN problem, distributed and parallel computation usually become inevitable to guarantee scalability. Effective data partitioning [? ?],load balancing [1] and minimizing the communication cost of the machines [5] have been studied for the (R)kNN problem so far. In this work, however, we leave the distributed version of our algorithm as a future work.

9 CONCLUSION

This paper introduces the problem of quantifying user search exposure, that is, finding the queries for which any of the user's posts is returned as a top-ranked result in a given search system. We proposed an efficient RkNN Growth Algorithm for computing the exposing queries as well as methods for ranking the queries in the resulting exposure sets to make the output user-friendly. To solve this ranking task, we moreover defined the concept of search exposure relevance, and studied it in a series of AMT surveys.

We believe there are a number of fascinating research questions that could be studied as an extension to the work presented in this paper. On the generation side, efficient stream processing of search exposure requests, including parallel computation, caching and request partitioning would be necessary in a real-world deployment. On the usability and ranking side, further understanding of exposure relevance, designing better ranking methods, studying the exposure under different search models, incorporating the likelihoods of queries being asked to the overall setup, or detecting exposure in black-box systems, are only a few of such extension possibilities. Finally, further investigating layman perceptions regarding search exposure as well as developing the expert understanding of the possible threats would give us a better grip of this newly defined privacy question.

REFERENCES

[1] Fuat Basık, Buğra Gedik, Hakan Ferhatosmanoğlu, and Mert Emin Kalender. 2015. S ˆ 33-TM: scalable streaming short text matching. *The VLDB JournalfiThe International Journal on Very Large Data Bases* 24, 6 (2015), 849–866.

[2] Holger Bast, Debapriyo Majumdar, Ralf Schenkel, Martin Theobald, and Gerhard Weikum. 2006. Io-top-k: Index-access optimized top-k query processing. In *Proceedings of the 32nd international conference on Very large data bases*. VLDB Endowment, 475–486.

[3] Joanna Asia Biega, Krishna P Gummadi, Ida Mele, Dragan Milchevski, Christos Tryfonopoulos, and Gerhard Weikum. 2016. R-susceptibility: An IR-centric approach to assessing privacy risks for users in online communities. In *SIGIR '16*.

[4] Andrei Z Broder, David Carmel, Michael Herscovici, Aya Soffer, and Jason Zien. 2003. Efficient query evaluation using a two-level retrieval process. In *Proceedings of the twelfth international conference on Information and knowledge management*. ACM, 426–434.

[5] B Barla Cambazoglu, Enver Kayaaslan, Simon Jonassen, and Cevdet Aykanat. 2013. A term-based inverted index partitioning model for efficient distributed query processing. *ACM Transactions on the Web (TWEB)* 7, 3 (2013), 15.

[6] David Carmel, Doron Cohen, Ronald Fagin, Eitan Farchi, Michael Herscovici, Yoelle S Maarek, and Aya Soffer. 2001. Static index pruning for information retrieval systems. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 43–50.

[7] Lisi Chen and Gao Cong. 2015. Diversity-Aware Top-k Publish/Subscribe for Text Stream. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. ACM, 347–362.

[8] Tobias Emrich, Hans-Peter Kriegel, Peer Kröger, Johannes Niedermayer, Matthias Renz, and Andreas Züfle. 2015. On reverse-k-nearest-neighbor joins. *GeoInformatica* 19, 2 (2015), 299–330.

[9] Ronald Fagin, Amnon Lotem, and Moni Naor. 2003. Optimal aggregation algorithms for middleware. *Journal of computer and system sciences* 66, 4 (2003), 614–656.

[10] Mainack Mondal, Peter Druschel, Krishna P Gummadi, and Alan Mislove. 2014. Beyond access control: Managing online privacy via exposure. In *Proceedings of the Workshop on Usable Security, USEC*, Vol. 14.

[11] Mainack Mondal, Johnnatan Messias, Saptarshi Ghosh, Krishna P Gummadi, and Aniket Kate. 2016. Forgetting in Social Media: Understanding and Controlling Longitudinal Exposure of Socially Shared Data. In *Twelfth Symposium on Usable Privacy and Security (SOUPS 2016)*.

[12] Edleno Silva de Moura, Celia Francisca dos Santos, Altigran Soares da Silva, Pavel Calado, Mario A Nascimento, and others. 2008. Locality-based pruning methods for web search. *ACM Transactions on Information Systems (TOIS)* 26, 2 (2008), 9.

[13] Linh Thai Nguyen. 2009. Static index pruning for information retrieval systems: A posting-based approach. In *SIGIR 2009 Workshop on Large-Scale Distributed Information Retrieval*. 25–32.

[14] Parikshit Ram and Alexander G Gray. 2012. Maximum inner-product search using tree data-structures. *arXiv preprint arXiv:1202.6101* (2012).

[15] Roman Schlegel, Apu Kapadia, and Adam J Lee. 2011. Eyeing your exposure: quantifying and controlling information sharing for improved privacy. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*.

[16] Reza Shokri, George Theodorakopoulos, George Danezis, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. 2011. Quantifying location privacy: the case of sporadic location exposure. In *International Symposium on Privacy Enhancing Technologies Symposium*.

[17] Akrivi Vlachou, Christos Doulkeridis, Yannis Kotidis, and Kjetil Norvag. 2011. Monochromatic and bichromatic reverse top-k queries. *IEEE Transactions on Knowledge and Data Engineering* 23, 8 (2011), 1215–1229.

[18] Man Lung Yiu and Nikos Mamoulis. 2007. Reverse nearest neighbors search in ad hoc subspaces. *IEEE Transactions on Knowledge and Data Engineering* 19, 3 (2007), 412–426.